# Lógica Aplicada a Computação

Gabriel Teixeira Patrício - 20170170889

Rennan Weslley da Silva Costa - 11515025

26/09/2017

A fórmula utilizada foi:
$(P \rightarrow Q) \wedge (P \rightarrow R) \leftrightarrow (Q \rightarrow R) \vee (Q \rightarrow P)$

Tabela verdade:

| P | Q | R | $(P \rightarrow Q) \wedge (P \rightarrow R) \leftrightarrow (Q \rightarrow R) \vee (Q \rightarrow P)$ |
|---|---|---|---|
| F | F | F | T |
| F | F | T | T |
| F | T | F | F |
| F | T | T | T |
| T | F | F | F |
| T | F | T | F |
| T | T | F | F |
| T | T | T | T |

Para adaptar o código à fórmula escolhida no programa 1 apenas a função **valor_formula()** foi alterada. Para bicondicional ($\leftrightarrow$) a função **BIMP()** foi por nós definida (BIMP(b1, b2) (b1 == b2)) como uma função que retorna **true** caso os valores passados por parâmetro sejam iguais e retorna **false** para caso sejam diferentes.

Para adaptar o programa 2 à fórmula mudamos o valor do ponteiro *f, do tipo **Formula** e localizado na função main, para a fórmula por nós escolhida através das funções disponibilizadas no código.

Para definir se a fórmula é **satisfatória** e se é **tautologia**, foi definido duas variáveis, **satis** e **taut**, uma representando se a fórmula é satisfatória e outra se é tautologia, respectivamente. O programa inicia assumindo que a fórmula não é satisfatória (satis = false) e que é uma tautologia (taut = true), caso haja pelo menos uma interpretação que resulte em **False**, já confirma-se que a fórmula não é **tautologia** (taut = false), e se pelo menos uma interpretação resultar em **True**, já confirma-se que a fórmula é satisfatória. O programa também informa se a fórmula é **negação**, já que se uma fórmula H é satisfatória, logo H não é negação, e se H não é satisfatória, isso implica que H é negação.

Para aplicar a nossa abordagem, foi necessário uma adaptação ao código e à lógica utilizada por cada um dos dois programas. A mudança semântica foi a mesma, mas a alteração no código-fonte em si foi diferente em cada um dos casos. Alterações feitas:

- Programa 1: Definimos as variáveis **satis** e **taut** como globais. A principal alteração foi feita na função **mostra_tabela()** onde é testado o valor da interpretação atual, caso a interpretação resulte em **true** a variável satis é setada para true (satis = true) e caso a interpretação resulte em **false** a variável taut é setada para false (taut = false). Na função main, após a exibição da tabela, foi adicionado um **if** e **else** para cada uma das duas variáveis onde são testados os seus valores. Na exibição, caso satis seja true é exibido a mensagem

que a fórmula é satisfatória e não é negação, e caso satis seja false é exibido uma mensagem que a fórmula não é satisfatória, logo é negação. O mesmo se aplica para o teste de valor da variável taut.

- Programa 2: Temos funções totalmente distintas mas de mesma funcionalidade com relação ao Programa 1. Adicionamos as variáveis satis e taut como atributos à struct **Formula**, em seguida modificamos a função **cria_formula()** adicionando as linhas que inicializam os valores satis e taut da fórmula passada por parâmetro. Também modificamos a função **mostra_tabela()** como fizemos no Programa 1, caso a interpretação resulte **true** a variável satis é setada para true, e caso resulte **false** a variável taut é setada para false. Igual ao Programa 1, na função main, após a exibição da tabela, adicionamos **if** e **else** para cada uma das duas variáveis onde são testados os seus valores. A exibição também não diferencia muito do programa anterior, caso satis da fórmula ***f** seja true é exibido a mensagem que a fórmula é satisfatória e não é negação, e caso satis da fórmula seja false é exibido uma mensagem que a fórmula não é satisfatória, logo é negação. O mesmo se aplica para o teste de valor da variável taut da fórmula ***f**.